University of Stuttgart

Reinforcement Learning

Introduction

Daniel Hennes

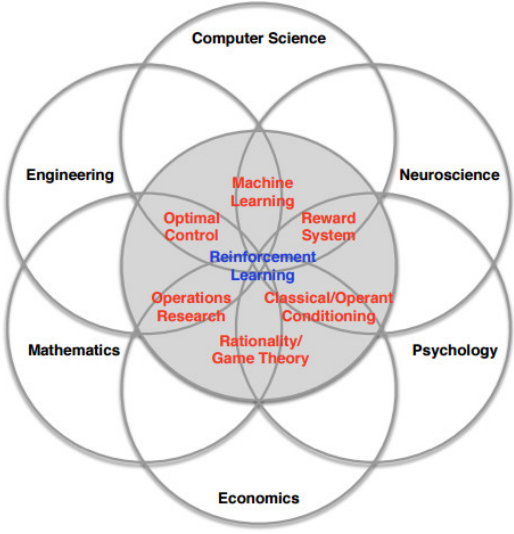
17.04.2017

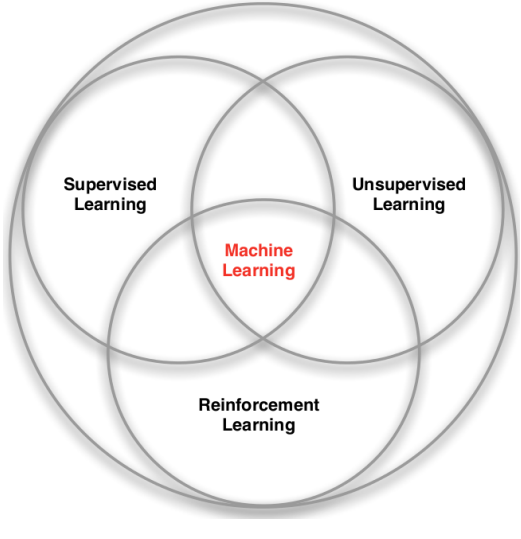
University Stuttgart - IPVS - Machine Learning & Robotics

What is reinforcement learning?

- General-purpose framework for *decision-making*
- Autonomous agent that *interacts* with its environments
  Learning through interaction
- Improving over time through *trial & error*
- **Agent** with the capacity to **act**
- Each **action** influences the future state
- Success is measured by a scalar **reward** signal
- Goal: **select actions to maximise future reward**

Many slides adapted from R. Sutton's course, David Silver's course, as well as previous RL courses given at U. of Stuttgart by M. Toussaint, H. Ngo, and V. Ngo.


Computer Science
Engineering
Neuroscience
Machine Learning
Optimal Control
Reward System
Reinforcement Learning
Operations Research
Classical/Operant Conditioning
Rationality/ Game Theory
Mathematics
Psychology
Economics


Supervised
Learning
Unsupervised
Learning
Machine
Learning
Reinforcement
Learning

The term “reinforcement learning”

- The term *“Reinforcement learning”* may refer to
  - a type of **problem**
  - the class of **solution methods** that work well on RL problems
  - the **research field** that studies RL problems and RL methods

It is important not to confuse the first two!

Characteristics of reinforcement learning

What makes reinforcement learning differnet from other machine learning paradigms?

- There is no supervisor, only a *reward* signal
- Feedback is (often) delayed, non instantaneous
- Time really matters (sequential, non i.i.d data)
- Agent's actions affect the subsequent data it receives

Examples of reinforcement learning

- **Fly stunt manoeuvres with a RC helicopter**
- **Learn to flip pancakes**
- Play boardgames (e.g., Backgammon, Go, Chess)
- Manage investment portfolios
- **Play Atari games at super human level**
- **Learning to walk**

Rewards

- A *reward* R_t is a scalar feedback signal
- *Only* feedback provided to the agent,
  no explicit teacher
- May indicate how well agent's last action was
- The agent's job is to maximise its expected cumulative reward
  over some (possibly) infinite horizon
- **Examples:**
  - winning or loosing a game (e.g., Backgammon, Go, . . .)
  - increaseing/decreasing score (e.g., video games)
  - earning/loosing money (e.g., portfolio managment)
  - following a desired trajectory vs. crashing (e.g., robotic control)
  - . . .

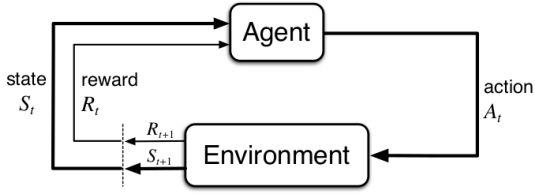
Can we describe *all* goals by the maximization of expected cumulative reward?

Sequential decision making

- Goal:*select actions to maximise total future reward*
- Actions may have long term consequences
- Reward may be delayed
- E.g., it may be better to sacrifice immediate reward to gain more long-term reward
- Examples:
  - A financial investment (may take months to mature)
  - Refuelling a helicopter (might prevent a crash in several hours)
  - Blocking opponent moves (might help winning chances many moves from now)

Interaction loop

- At each step t, the agent:
  - receives observation O_t
  - receives scalar reward R_t
  - executes action A_t
- The environment:
  - receives action A_t
  - emits observation O_{t+1}
  - emits scalar reward R_{t+1}
- t increments at environment step



History and state

- The **history** is the sequence of observations, actions, rewards

$$H_t = O_1, R_1, A_1, \ldots, A_{t-1}, O_t, R_t$$

- i.e. all observable variables up to time t
- i.e. the sensorimotor stream of a robot or embodied agent
- What happens next depends on the history:
  - The agent selects actions
  - The environment selects observations/rewards
- **State** is the sufficient information used to determine what happens next
- Formally, the state is a function of the history:

$$S_t = f(H_t)$$

Information state

An **information state** (a.k.a. *Markov* state) contains all useful information from the history.

A state S_t is Markov if and only if

$$\Pr\{S_{t+1} \mid S_t\} = \Pr\{S_{t+1} \mid S_1, \ldots, S_t\}$$

The future is independent of the past given the present

$$H_{1:t} \to S_t \to H_{t+1:\infty}$$

- Once the state is known, the history may be thrown away, i.e. the state is a sufficient statistic of the future
- The history H_t is Markov

Fully and partially observable environments

- If the agent directly observes the Markov state,
  we call the interaction model a **Markov Decision Process** (MDP)
- If the agent indirectly observes the environment state,
  we call it a **Partially Observable Markov Decision Process** (POMDP)
- Many (if not all) real world examples are POMDPs
- Examples:
  - a robot with camera vision isn't told its absolute location
  - a trading agent only observes current prices
  - a poker playing agent only observes public cards

Building blocks of RL agents

- **Policy:** agent's behavior
- **Value function:** how good is (*a given action in*) a given state?
- **Model:** agent's representation of the environment

Policy

- Defines the agent's behavior
- Maps from state to action
- **Deterministic policy:** $a = \pi(s)$
- **Stochastic policy:** $\pi(a \mid s) = \Pr\{A_t = a \mid S_t = s\}$

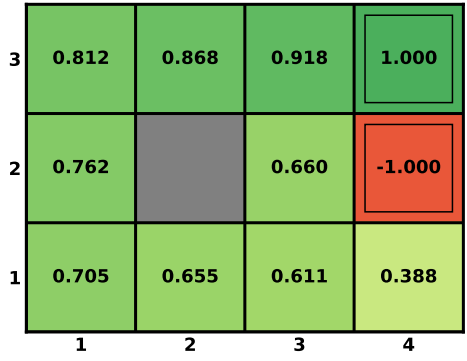
Value function

- Value function is a *prediction of future reward*

$$v_\pi(s) = \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots \mid S_t = s]$$

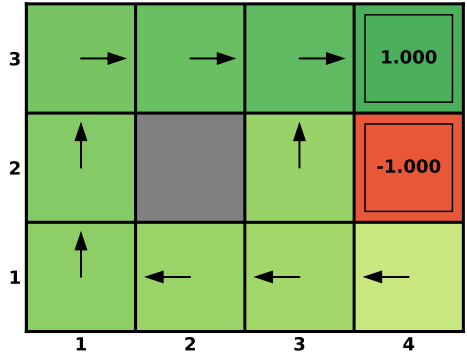
- Used to evaluate the goodness/badness of states
- And thus to select between actions

Example: grid world

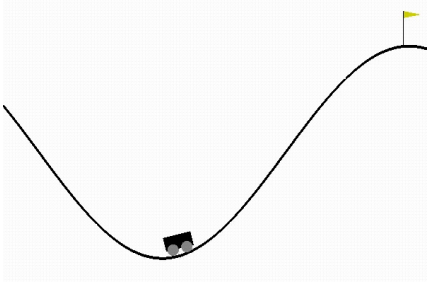


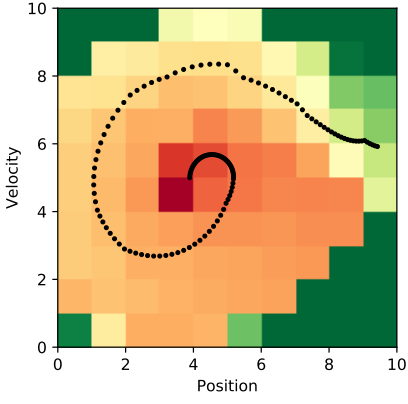
- Rewards: 0, $+1$, -1
- Actions: N, E, S, W
- States: agent's location

Example: grid world



Example: mountain car





Model

- A **model** predicts what the environment will do next
  - the next state s'
  - the next (immediate) reward r

$$p(s', r \mid s, a) = \Pr\{S_{t+1} = s', R_{t+1} = r \mid S_t = s, A_t = a\}$$

Many flavours of reinforcement learning

model-based $S_t, R_t, A_t, S_{t+1} \ldots \rightarrow p(s' \mid s, a), r(s, a, s') \rightarrow v(s) \rightarrow \pi(s)$

model-free

value-based $S_t, R_t, A_t, S_{t+1} \ldots \rightarrow q(s, a) \rightarrow \pi(s)$

policy-based $S_t, R_t, A_t, S_{t+1} \ldots \rightarrow \pi(s)$

actor-critic $S_t, R_t, A_t, S_{t+1} \ldots \rightarrow q(s, a), \pi(s)$

imitation learn. $\left\{(S_{1:T}, A_{1:T}, R_{1:T})^i\right\}_{i=1}^n \rightarrow \pi(s)$

learning | dynamic programming

Learning or planning?

- **Reinforcement Learning:**
  - the environment is (initially) unknown
  - the agent interacts with the environment
  - the agent improves its policy
- **Planning:**
  - a model of the environment is known
  - the agent performs computations with its model (without any actual interaction)
  - the agent improves its policy

Exploration vs. exploitation

- Reinforcement learning is *trial & error* learning
- The agent should discover a good policy
  - from its experiences of the environment
  - without losing too much reward along the way
- **Exploration** finds more information about the environment
- **Exploitation** exploits known information to maximise reward
- Examples:
  - **Dining:** go to your favorite restaurant vs. try something new
  - **Advertisment:** place a new advert vs. the most relevant
  - **Mars rover:** sample a new location vs. sample best so far
  - **Game playing:** play a new move vs. the move that worked in the past

Success of reinforcement learning

- **Games:**
  - Backgammon (Tesauro, 1994)
  - Deep RL playing Atari (2014)
  - AlphaGo (2016)
- **Operations research:**
  - Inventory Management (Van Roy, Bertsekas, Lee, & Tsitsiklis, 1996)
  - Dynamic Channel Allocation (e.g. Singh & Bertsekas, 1997)
  - Investment portfolio managment
  - Online advertisements
- **Robotics:**
  - Helicopter control (Ng 2003, Abbeel & Ng 2006)
  - Bi-pedal walking
  - Grasping

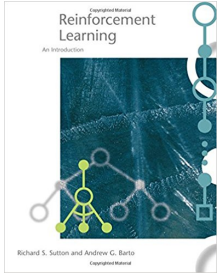
Admin

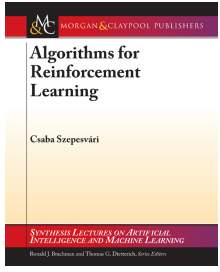
- **Lectures:** Tuesday, 17:30 - 19:00, room V38.03
- **Tutorials:**
  - Wednesday, 14:00 - 15:30, room 0.108
  - Wednesday, 15:45 - 17:15, room 0.447
- **Office hours:** by appointment
- **Communication:** website & mailing list
- **Contact:**
  - daniel.hennes@ipvs.uni-stuttgart.de
  - carola.stahl@ipvs.uni-stuttgart.de
- **Website:**
  https://ipvs.informatik.uni-stuttgart.de/mlr/reinforcement-learning-ss-18/

Tutorials

- *Doing the exercises is crucial!*
- At the beginning of each tutorial:
  - sign into the list
  - mark which exercises you have (successfully) worked on
- Students are randomly selected to present their solutions
- You need to complete at least **50%** of the exercises to be allowed to the exam

Literature





- *Reinforcement Learning: An Introduction* (2nd ed.) by Richard Sutton and Andrew Barto http://incompleteideas.net/book/bookdraft2017nov5.pdf
- *Algorithms for Reinforcement Learning* by Cesba Szepesvari https://sites.ualberta.ca/~szepesva/papers/RLAlgsInMDPs.pdf

Announcements

- This week (tomorrow): no tutorials!
- Next week, lecture in room **V38.01**!